Efficient Content Distribution in semi-decentralized Peer-to-Peer-Networks

Outline

- Introduction
- Content Distribution
- The P2P-Paradigm
- Encountered Problems
- Solutions
- A System Sketch
- Conclusion
Content Distribution

why are we doing this?

Application Domains:

- Publish large files over the internet
  (for a distinguished group of people, the history)
- Groupware (Sharing e.g. scientific papers)
  (in a special interest group, the easy way)
- POF (Plain old file sharing)
  (using efficient data transfer!)
- P2P Media Streaming

Existing Systems

why aren’t we using one of the quadzillion systems?

Existing solutions

- Centralized systems
  (+) good manageability
  (-) load balancing, fault tolerance, scalability (large amounts of data)
- Decentralized (P2P-) systems
  (-) manageability, scalability (global requests)
  (+) fault tolerance, load balancing, autonomous users
The Peer-to-Peer-Paradigm

servers? hierarchy?

A Peer-to-Peer-Network:
- a network of equal servants, running the same application (-type)
- each servant performs (is able to perform) the same task
→ There are no dedicated servers in a P2P-Network
- each servant may act as a server or a client in different communication relations

Peer-to-Peer-Networks

- P2P-Networks are „overlay-networks“ on the application-layer
→ implement their own routing mechanisms
Known Issues with P2P-Networks
Scalability, Integrity of the Network

- Application Level: the discovery problem
  - Search in unstructured decentralized networks
    (doesn’t scale, doesn’t search entire network \(\rightarrow\) is unreliable)

- Network Level: inefficient bandwidth use
  - Single source download
  - No information about geographical locality of a peer

- Fragmentation of the network:
  - Failing „hubs“

Issues with P2P-Networks
Publishing, Groups & Trust

- No way to „push“ data to a set of servants

- No group services
  - No consistent identification
  \(\rightarrow\) No authorization methods
  - No dedicated authentication service
Making P2P Scalable

Application level

Getting structure into the chaos:

- Introduce „supernodes“ as discovery services
- Servants connect to supernodes and publish own resources
- Restrict discovery to the supernodes
- Use message chains to avoid traffic
- Reconfiguring servants

Equal servants? With the same capabilities?
Making P2P Scalable

Network level

Introduction of relative locality:
- determine relative distance to peers
- register with "n" nearest discovery services
- retrieve resources from the nearest source
- try to obtain a complete replica "for the region"
  ➔ retrieve locally unavailable fragments from remote sources

Providing Group Services

- Open groups are easy to implement
  (simply use meta-information)
- Closed groups require additional features:
  - Identification
  - Authorization
  - Authentication
Providing Group Services

Authorization

Flat vs. role based authorization:
- Flat group authorization
  - All members are trusted equally
  - Impossible to exclude member from a group
- Role based group authorization
  - Creator of the group becomes its owner
  - Possibility to appoint substitute

Æ Role based group authorization needed

Authentication with no well-known security services

- Public key cryptography is a basis for trust
- Build a group repository containing the members
- Use trusted nodes → group service „supernodes“
- Groupnodes register with „n“ group services
- Schroeder-Needham Protocol to authenticate a listed peer

We need notification, key distribution and secure naming!
Conclusion

- The evolving system provides features for efficient and scalable content provisioning
- Implements secure group communication
- Is a basis for a simple middleware for extremely dynamic environments
Questions?

nothing here...